CS534 - Machine Learning

Written Homework Assignment 2

Author: Vy Bui
OSUID: 934370552
Email: buivy@oregonstate.edu

The School of Electrical Engineering and Computer Science
Oregon State University

This assignment covers Perceptron, Kernel methods, SVMs and Naive Bayes

1. (Subgradient) (2 pts) Consider the L_1 form function for $\mathbf{x} \in R^d$: $f(\mathbf{x}) = |\mathbf{x}|_1 = \sum_{i=1}^{d} |x_i|$. Show that $\mathbf{g} = [g_1, g_2, ..., g_d]^T$ is a subgradient of $f(\mathbf{x})$ at $\mathbf{x} = \mathbf{0}$ if every $g_i \in [-1, 1]$ using the definition of subgradient: g is a subgradient of $f(x)$ at x_0 if $\forall x$, $f(x) \geq f(x_0) + g^T(x - x_0)$

   Let $z = x_0 = 0$ to avoid notation confusion. First, let us consider the right side of the inequality R. $f(z) = 0$ because $z_i = 0, i \in [1, d]$, hence the right side reduces to

   $$R = \mathbf{0} + \mathbf{g}^T(\mathbf{x} - \mathbf{z}) = g_1 x_1 + g_2 x_2 + ... + g_d x_d$$

   The left side can be calculated as follows

   $$f(x) = |x_1| + |x_2| + ... + |x_d|$$

   Because $g_i \in [-1, 1]$, $g_i x_i \leq |x_i|$ for $i = 1...d$, therefore $f(x) \geq f(z) + \mathbf{g}^T(\mathbf{x} - \mathbf{z})$, which shows that g is a subgradient of $f(x)$ at $x = 0$.

2. (Perceptron) (2 pts) Consider the following argument. We know that the number of steps for the perceptron algorithm to converge for linearly separable data is bounded by $(\frac{D}{\gamma})^2$. If we multiple the input $\mathbf{x}$ by a small constant α, which effectively reduces the bound on $|\mathbf{x}|$ to $D' = \alpha D$, we can reduce the upper bound to $(\alpha\frac{D}{\gamma})^2$. Is this argument correct? Why?

   This argument is incorrect. Because the scaler α not only affects "the radius of data" D, but also affects the maximum margin γ, which is the distance between the separating hyperplane to the closest x_i. As a result, the upper bound of the number of steps becomes $(\frac{\alpha D}{\beta\gamma})^2$, where β is the effect of scaling x on γ.

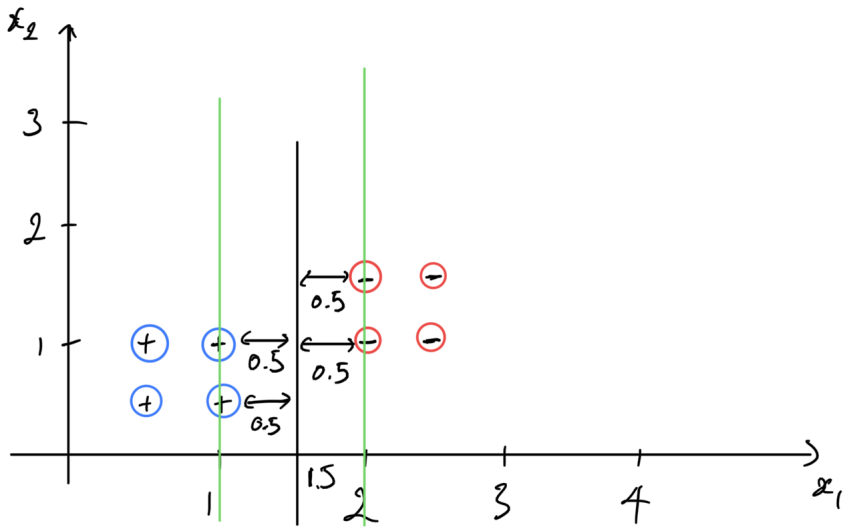
3. (Kernel or not). (9 pts) In the following problems, suppose that K, K_1 and K_2 are kernels with feature maps ϕ, ϕ_1 and ϕ_2. For the following functions $K'(x, z)$, state if they are kernels or not. If they are kernels, write down the corresponding ϕ in terms of ϕ, ϕ_1 and ϕ_2. If they are not kernels, prove that they are not.

   - (2 pts) $K'(\mathbf{x}, \mathbf{z}) = cK(\mathbf{x}, \mathbf{z})$ for $c > 0$.
     This is a valid kernel with the corresponding $\phi' = \sqrt{c}\phi$
   - (2 pts) $K'(\mathbf{x}, \mathbf{z}) = cK(\mathbf{x}, \mathbf{z})$ for $c < 0$.
     This is not a valid kernel because $\phi' = \sqrt{c}\phi$ is not defined when $c < 0$.
   - (2 pts) $K'(\mathbf{x}, \mathbf{z}) = c_1 K_1(\mathbf{x}, \mathbf{z}) + c_2 K_2(\mathbf{x}, \mathbf{z})$ for $c_1, c_2 > 0$.
     This is a valid kernel with the corresponding $\phi = \sqrt{c_1}\phi_1 + \sqrt{c_2}\phi_2$
   - (3 pts) $K'(\mathbf{x}, \mathbf{z}) = K_1(\mathbf{x}, \mathbf{z})K_2(\mathbf{x}, \mathbf{z})$.
     This is a valid kernel with the corresponding $\phi = \phi_1\phi_2$

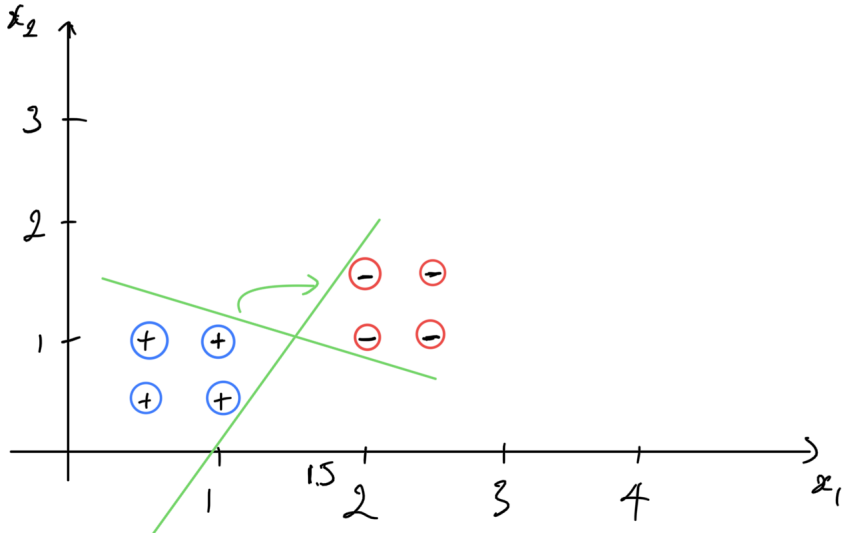
4. (Hard margin SVM) (5 pts) Apply linear SVM without soft margin to the following problem.

   a. (2pts) Please mark out the support vectors, the decision boundary ($w_1x_1 + w_2x_2 + b = 0$) and $w_1x_1 + w_2x_2 + b = 1$ and $w_1x_1 + w_2x_2 + b = -1$. You don't need to solve the optimization problem for this, you should be able to eyeball the solution and find the linear separator with the largest margin.

   As shown in the following figure, the black line $x_1 = 1.5$ is the linear separator with the largest margin equal 0.5. The support vectors are the points through which the green lines $x_1 = 1$ and $x_1 = 2$ go.



   Any change in the slope of the linear separator will decrease the margin, as shown in the following figure.



   b. (3 pts) Please solve for w_1, w_2 and b based on the support vectors you identified in (a). Hint: the support vectors would have functional margin = 1.

   The functional margin for data point (1.5,0) on the decision boundary, support vectors (1,1) and (1,2) can be calculated as follows

   $1.5w_1 + b = 0$

   $w_1 + w_2 + b = 1$

   $2w_1 + w_2 + b = -1$

   Solving this linear system results in $w_1 = -2, w_2 = 0, b = 3$.

5. L_2 SVM (10 pts)
Given a set of training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, where $y_i \in \{1, -1\}$ for all i. The following is the primal formulation of L_2 SVM, a variant of the standard SVM obtained by squaring the slacks.

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \ & \frac{1}{2}\mathbf{w}^T\mathbf{w} + c\sum_{i=1}^{N}\xi_i^2 \\ \text{s.t. } & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i,\ i \in \{1, \cdots, N\} \\ & \xi_i \geq 0,\ i \in \{1, \cdots, N\} \end{aligned}$$

a. (2pts) Show that removing the second constraint $\xi_i \geq 0$ will not change the solution to the problem. In other words, let $(\mathbf{w}^*, b^*, \xi^*)$ be the optimal solution to the problem without this set of constraints, show that $\xi_i^* \geq 0,\ \forall i \in \{1, \cdots, N\}$. (Hint: use proof by contradiction by assuming that there exists some $\xi_i^* < 0$.)

Assume that there exists some optimal solution $(\mathbf{w}^*, b^*, \xi^*)$ with $\xi_i^* < 0, \forall i \in \{1, ..., N\}$. Then the solution $(\mathbf{w}^*, b^*, \mathbf{0})$ will also satisfy the constraint $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i$ and make the objective function smaller. This contradicts with the assumption that the solution is optimal, therefore, there exists no optimal solution with $\xi_i^* < 0, \forall i \in \{1, ..., N\}$. Hence, removing it from the list of constraints does not change the solution to the problem.

b. (2 pts) After removing the second set of constraints, we have a simpler problem with only one set of constraints. Now provide the lagrangian of this new problem.

$$\mathcal{L}(w, b, \alpha, \xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + c\sum_{i=1}^{N}\xi_i^2 - \sum_{i=1}^{N}\alpha_i[y_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i]$$

s.t. $\alpha_i \geq 0$ for $i = 1, ..., m$.

c. (6pts) Derive the dual of this problem. How is it different from the standard SVM with hinge loss? Which formulation is more sensitive to outliers?
Substituting $\mathbf{w} = \sum_{i=1}^N \alpha_i y_i \mathbf{x}_i$ into $\mathcal{L}$ results in

$$\mathcal{L}(\alpha) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i y_i \mathbf{x}_i)^T(\alpha_j y_j \mathbf{x}_j) + \sum_{i=1}^{N}\frac{\alpha_i}{\xi_i}\xi_i^2 - \sum_{i=1}^{N}\alpha_i[y_i((\sum_{j=1}^{N}\alpha_j y_j \mathbf{x}_j)^T\mathbf{x}_i + b) - 1 + \xi_i]$$

$$= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j + \sum_{i=1}^{N}\alpha_i\xi_i - b\sum_{i=1}^{N}\alpha_i y_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i$$

$$= \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$$

The loss formulation for the dual problem is more sensitive to outliers because the degrees of x and y in it are higher than those in the formulation of Hinge loss.

6. (Naive Bayes Classifier) (6 pts) Consider the following training set:

A	B	C	Y
0	1	1	0
1	1	1	0
0	0	0	0
1	1	0	1
0	1	0	1
1	0	1	1

(a) (2 pts) Learn a Naive Bayes classifier by estimating all necessary probabilities (there should be 7 independent probabilities to be estimated in total).

First, the prior can be calculated as $P(y=1)=0.5$ *and* $P(y=0)=0.5$.

Second, the distribution of A,B,C given y can be computed as

$P(A=0|y=1)=1/3, P(A=1|y=1)=2/3$

$P(B=0|y=1)=1/3, P(B=1|y=1)=2/3$

$P(C=0|y=1)=2/3, P(C=1|y=1)=1/3$

Finally,

$P(A=1,B=0,C=0)=\sum_j(\prod_i P(x_i|y=j)P(y=j))$

$=0.5\times 1/3\times 1/3\times 1/3+0.5\times 2/3\times 1/3\times 2/3=\frac{5}{54}$

(b) (3 pts) Compute the probability $P(y=1|A=1,B=0,C=0)$.

$P(y=1|A=1,B=0,C=0)=\frac{P(A=1,B=0,C=0|y=1)P(y=1)}{P(A=1,B=0,C=0)}=\frac{2/3\times 1/3\times 2/3\times 0.5}{5/54}=\frac{4}{5}$

(c) (1 pts) Suppose we know that the three features A, B and C are independent from one another, can we say that the Naive Bayes assumption is valid? (Note that the particular data set is irrelevant for this question). If your answer is yes, please explain why; if you answer is no please give an counter example.

The assumption is not valid because the independence of between A,B, and C is not conditional independence, which is required for Naive Bayes assumption.

Counter example:

Independence:

$P(A=0,B=0,C=0|y=1)=P(A=0)P(B=0)P(C=0)=1/2\times 2/6\times 1/2=1/12$

Conditional independence:

$P(A=0,B=0,C=0|y=1)=P(A=0|y=1)P(B=0|y=1)P(C=0|y=1)=1/3\times 1/3\times 2/3=2/27$

7. (Naive Bayes learns linear decision boundary.) (6 pts) Consider a naive Bayes binary classifier with a set of binary features $x_1, x_2, ..., x_d$. Show that the Naive Bayes classifier learns a linear decision boundary $w_0+w_1x_1+w_2x_2+...+w_dx_d=0$. Express the weights using the Naive Bayes parameters. Hint: consider the decision rule of predicting $y=1$ if $P(y=1|\mathbf{x})>P(y=0|\mathbf{x})$. This is equivalent to having a decision boundary defined by $\log\frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})}=0$.

The classifier will predict $y=1$ *if* $P(y=1|\mathbf{x})\geq P(y=0|\mathbf{x})\Leftrightarrow\frac{P(\mathbf{x}|y=1)P(y=1)}{P(\mathbf{x}|y=0)P(y=0)}\geq 1$ (1)

Applying the naive Bayes assumption, we have

$$P(\mathbf{x}|y)=\prod_{i=0}^{d}P(x_i|y) \quad (2)$$

Plugging (2) into (1) results in

$$\frac{P(y=1)}{P(y=0)}\prod_{i=0}^{d}\frac{P(x_i|y=1)}{P(x_i|y=0)}\geq 1 \quad (3)$$

To simplify notation, let $P(y=1)=p$, $P(x_i=1|y=1)=k_i$, *and* $P(x_i=1|y=0)=m_i$. *Because the features are binary, either 0 or 1, we can derive* $P(x_i|y=1)=k_i^{x_i}(1-k_i)^{1-x_i}$ *and* $P(x_i|y=0)=m_i^{x_i}(1-m_i)^{1-x_i}$.

Plugging these new notations to (3) we get

$$\frac{p}{1-p}\prod_{i=1}^{d}\frac{k_i^{x_i}(1-k_i)^{1-x_i}}{m_i^{x_i}(1-m_i)^{1-x_i}}\geq 1$$

$$\Leftrightarrow (\frac{p}{1-p}\prod_{i=0}^{d}\frac{1-k_i}{1-m_i})\prod_{i=0}^{d}(\frac{k_i(1-m_i)}{m_i(1-k_i)})^{x_i} \geq 1$$

$$\Leftrightarrow log(\frac{p}{1-p}\prod_{i=0}^{d}\frac{1-k_i}{1-m_i}) + \sum_{i=0}^{d} x_i log(\frac{k_i(1-m_i)}{m_i(1-k_i)}) \geq 0 \quad (4)$$

Because the log term does not contain any x_i, it can be seen as a constant c. The log inside of the summation is actually w_i. Hence, (4) becomes

$$c + \sum_{i=0}^{d} x_i w_i \geq 0$$

This shows that the classifier learns a linear decision boundary with w equal to the log term in the summation in (4).